gossple.sh(1) gossple.sh User Manual

Bryan Beresford-Smith, Andrew Bromage,
Thomas Conway, Jeremy Wazny, Justin Zobel

April 18, 2012

NAME

gossple.sh - a front end script for doing simple assembly with Gossamer.

Version 1.2.1

SYNOPSYS

gossple.sh [options] file ...

OPTIONS

-B *buffer-size* amount of memory in GB to use for buffering. Should be somewhat less than main memory size. (default 2GB.)

-c *coverage-estimate* estimated k-mer coverage. Used during pair resolution to determine whether sequences are unique. (default automatic estimation.)

-D dry run. Form the commands, but do not execute them (usually used with -v).

-h help. Print a help message and exit.

-k *k-mer-size* k-mer size for composing the *de Bruijn* graph. (default 27.)

-m *insert-size* insert size for mate pair data. (See below.)

-o *file-name* name of a file to write the contigs into. (default 'contigs.fa')

-p *insert-size* insert size for paired end data. (See below.)

-T *num-threads* number of worker threads to use for parallel operations. (default 4.)

-t *coverage-threshold* the coverage at which to trim low frequency edges. (default auto.)

-V print verbose progress output from the goss commands.

-v print the goss command lines being used.

-w *working-dir* path to a working directory for graphs and other working files.

—**tmp-dir *temporary-dir*** path to a directory for temporary files which are removed after each Gossamer run.

DESCRIPTION

The gossple.sh script is a simple shell script front end to goss which invokes goss multiple times to effect a default assembly pipeline.

The default pipeline uses all the data (single ended, and paired) for building the initial graph, which is then processed to eliminate errors by trimming low frequency edges, and detecting structural motifs (*tips* and *bubbles*) which correspond to errors. Any paired data are then used to disambiguate conflated paths in the graph, and contigs are read off.

Input Files

gossple.sh does filename pattern matching to determine the kind of input data. Files ending in '.fa[.gz|.bz2]' or '.fasta[.gz|.bz2]' are recognised as FASTA files. Files ending in '.fq[.gz|.bz2]' or '.fastq[.gz|.bz2]' are recognised as FASTQ files. In both cases, files are decompressed on the fly if necessary[1].

Paired data

gossple.sh allows unpaired, paired end and mate pair data all to be used in a single assembly. Files containing paired end data are introduced with "-p *insert-size*" and following pairs of files are treated as containing corresponding reads. Likewise, mate pair data are introduced with "-m *insert-size*". Where multiple libraries are used, the command line should be formed:

```
gossple.sh [options] file-name* {-p *size* file-name* | -m *size* file-name*}*
```

[1] Release 1.1.0 has bzip2 decompression disabled due to bugs in third party libraries.

All the files containing single ended data should be given before the first '-p' or '-m'. Multiple groups of paired data may be given, as in

```
gossple.sh [options] -p 200 a_1.fq a_2.fq -p 600 b_1.fq b_2.fq -m 3000
c_1.fq c_2.fq
```

Pair resolution is performed in the order in which the groups are given, so it is recommended that they be given in order of insert size - smallest to largest.

ENVIRONMENT VARIABLES

The GOSSAMER environment variable, if set, is used to specify the goss executable. If not set, the command "goss" is executed from the environment in the usual way.

Publication

We respectfully ask that if you publish results created with Gossamer, please reference the paper:

Thomas C Conway, Andrew J Bromage, "Succinct data structures for assembling large genomes", Bioinformatics, 2011 vol. 27 (4) pp. 479–86 **GossamerPaper**